คำสั่งวนซ้ำ

นายณัชพล กาฬภักดี
ครู ชำนาญการพิเศษ
โรงเรียนบ่อกรุวิทยา

สำนักงานเขตพื้นที่การศึกษามัธยมศึกษา เขต 9

ทำไมต้องวนซ้ำ

- เพื่อทำการคำนวณค่า เปรียบเทียบค่า ที่อยู่ในรอบ
- เพื่อเป็นการสร้างตัวแปรให้น้อยลง
- อื่น ๆ

การกำหนดค่าตัวนับ

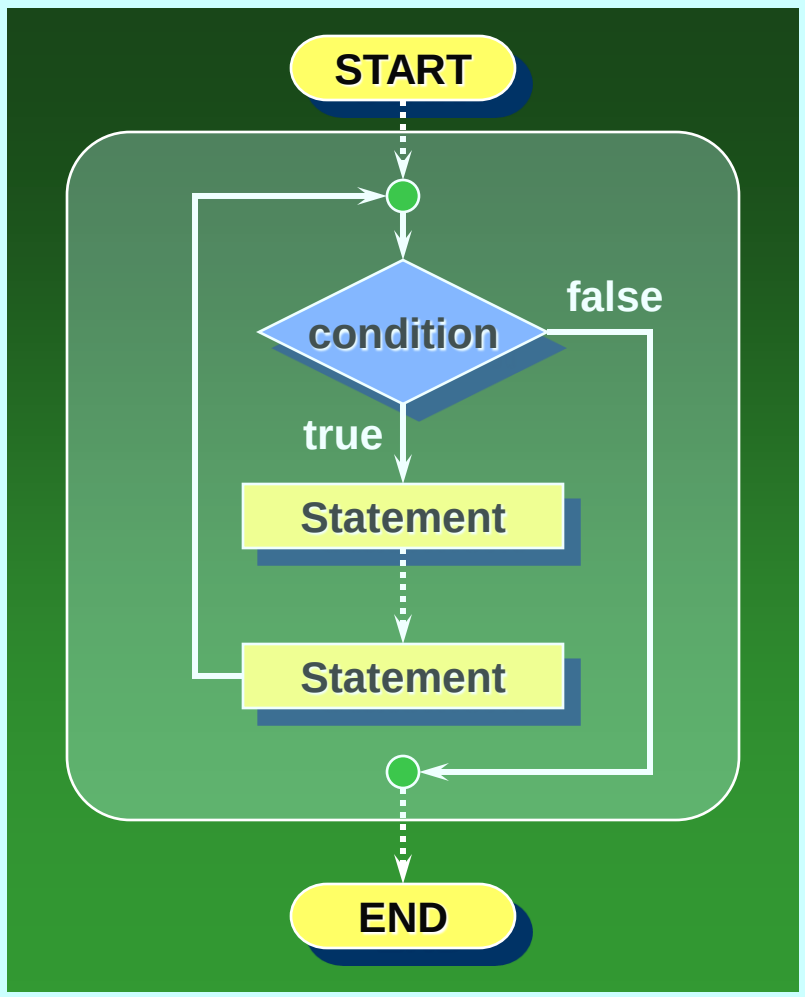
ตัวอย่าง

i++	=	i = i+1
i--	=	i = i-1
i+=5	=	i = i+5
i-=5	=	i = i-5

โครงสร้าง while ลูป

```
while (condition)
{
    stmt1;
    stmt2;
    :
    stmtN;
}
```

- วนทำคำสั่ง ***stmt1*** ถึง ***stmtN*** ตราบเท่าที่ ***condition*** เป็นจริง



ลูปวนนับ (Counting Loop)

- หากพิจารณาโครงสร้างของลูปที่ใช้ในโปรแกรมส่วนใหญ่มักจะเป็นลูปแบบวนนับ
- ลูปวนนับจะมีส่วนประกอบดังตัวอย่างต่อไปนี้เสมอ

```
int i, sum = 0;
i = 1;
while (i <= 10)
{
  sum = sum + i;
  i = i + 1;
}
printf("Sum = %d\n", sum);
```

ตัวแปรที่ใช้นับ

ส่วนกำหนดค่าเริ่มต้น

เงื่อนไขของตัวนับ

คำสั่งที่ถูกทำซ้ำ

การปรับค่าตัวนับ

ตัวอย่างโครงสร้าง while ลูป

```
#include <stdio.h>
int main()
{
  int i=1;
  while(i<=10)
  {
    printf("Hello %d\n",i);
    i++;
  }
  getch();
  return 0;
}
```

while1.c

```
Hello 1
Hello 2
Hello 3
Hello 4
Hello 5
Hello 6
Hello 7
Hello 8
Hello 9
Hello 10
```

โครงสร้าง while ลูป(INFINITY LOOP)

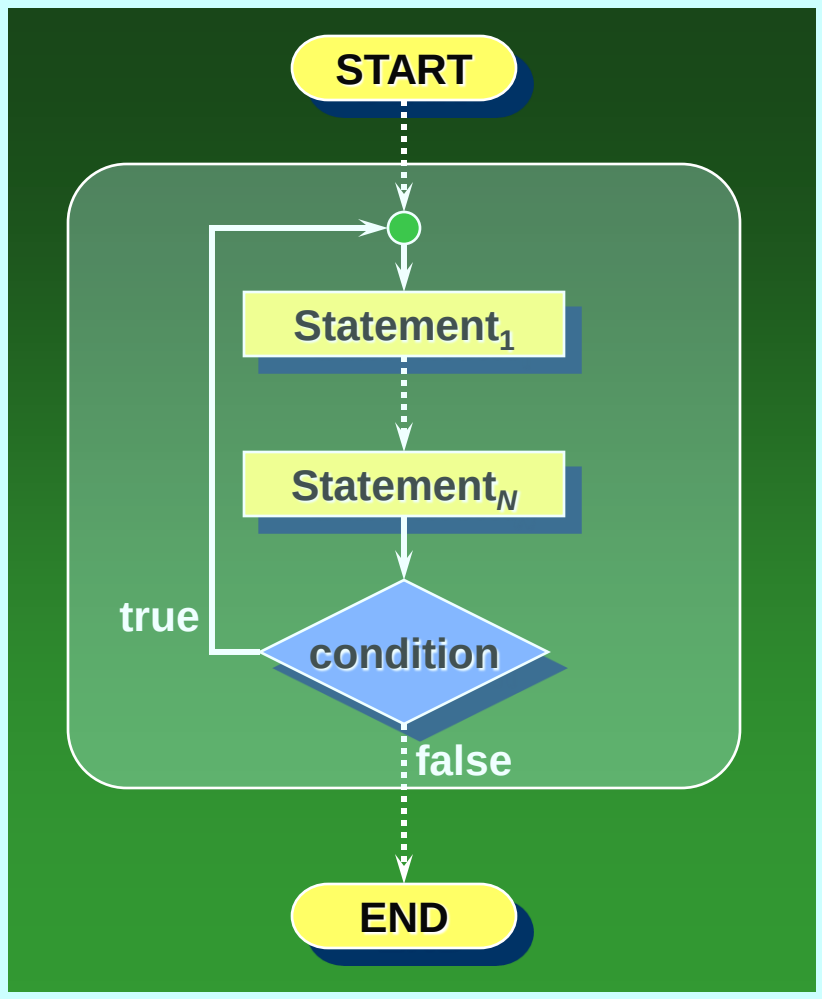
```
#include <stdio.h>
int main()
{
  int i=1;
  while(1)
  {
    printf("Hello %d\n",i);
     if(i==10)
       break;
  i++;
 }
 getch();
 return 0;
}
```

while2.c

```
Hello 1
Hello 2
Hello 3
Hello 4
Hello 5
Hello 6
Hello 7
Hello 8
Hello 9
Hello 10
```

โครงสร้าง do...while ลูป

```
do
{
    stmt1;
    stmt2;
    :
    stmtN;
} while (condition);
```



- ทำคำสั่ง *stmt1...stmtN* และวนทำซ้ำอีกตราบเท่าที่ *condition* ยังคงเป็นจริง
- นั่นคือ *stmt1...stmtN* จะถูกกระทำอย่างน้อยหนึ่งครั้ง

ตัวอย่าง do...while ลูป

```
#include <stdio.h>
int main()
{
  int i=1;
  do
  {
    printf("Hello %d\n",i);
    i++;
  }
  while(i<=10);
 getch();
 return 0;
}
```

dowhile1.c

```
Hello 1
Hello 2
Hello 3
Hello 4
Hello 5
Hello 6
Hello 7
Hello 8
Hello 9
Hello 10
```

โครงสร้าง for ลูป

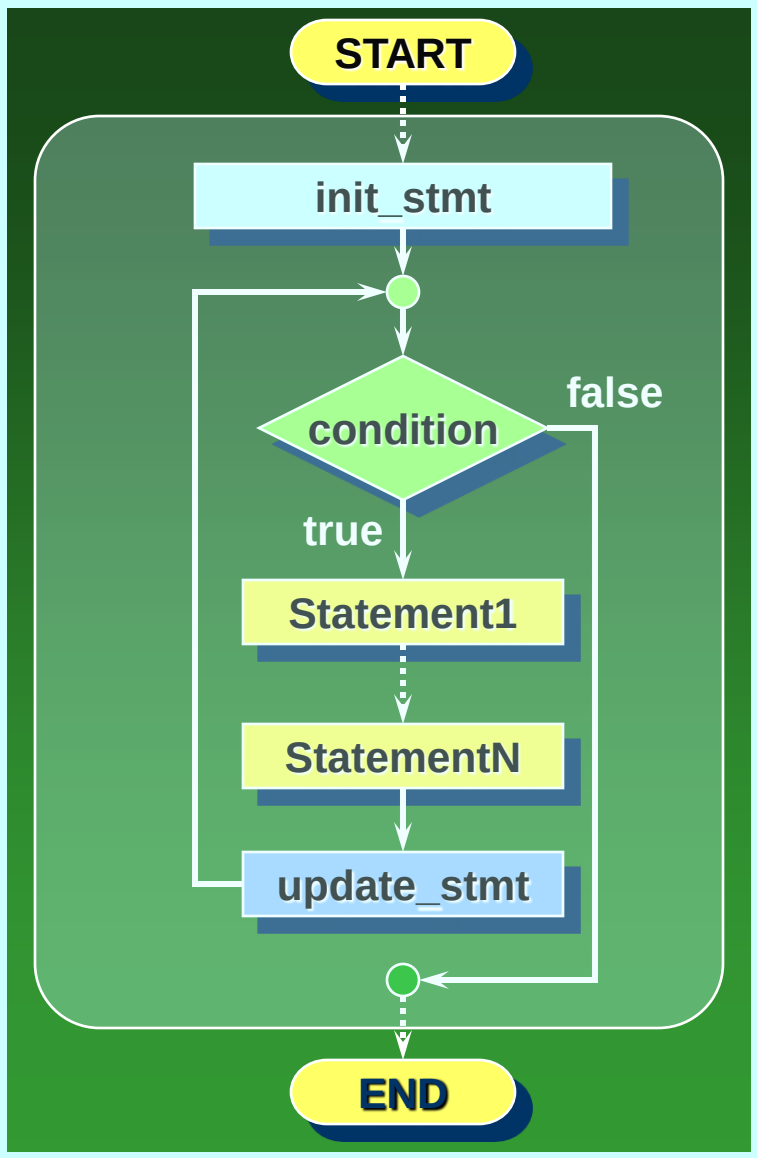
- เป็นโครงสร้างที่ให้ความสะดวกในการเขียนลูปวนนับ

```
for (init_stmt; condition; update_stmt)
{
    statement1;
    statement2;
    :
    statementN;
}
```

- การทำงาน
  1. ทำคำสั่ง *init_stmt* หนึ่งครั้ง
  2. ถ้า *condition* เป็นจริง ทำคำสั่ง ***statement1...statementN***
  3. ทำคำสั่ง *update_stmt* จากนั้นกลับไปทำข้อ 2

การทำงานของ for ลูป

```
for (init_stmt; condition; update_stmt)
{
    statement1;
    statement2;
    :
    statementN;
}
```



ตัวอย่าง for ลูป

```
#include <stdio.h>
int main()
{
  int i;
  for(i=1;i<=10;i++)
  {
    printf("Hello %d",i);
    printf("\n");
  }
 getch();
 return 0;
}
```

for1.c

```
Hello 1
Hello 2
Hello 3
Hello 4
Hello 5
Hello 6
Hello 7
Hello 8
Hello 9
Hello 10
```